Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Discontinued at Digi-Key
Core Processor	ARM® Cortex®-M3
Core Size	32-Bit Single-Core
Speed	32MHz
Connectivity	EBI/EMI, I²C, IrDA, SmartCard, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, LCD, POR, PWM, WDT
Number of I/O	56
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	1.98V ~ 3.8V
Data Converters	A/D 8x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-QFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/efm32g30f128-qfn64t

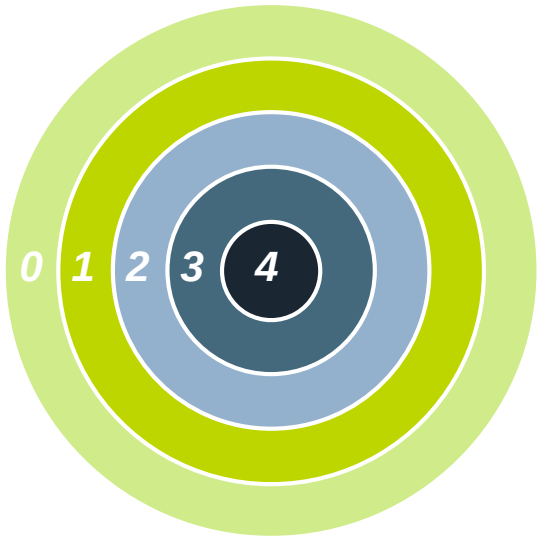
1 Energy Friendly Microcontrollers

1.1 Typical Applications

The EFM32G Gecko is the ideal choice for demanding 8-, 16-, and 32-bit energy sensitive applications. These devices are developed to minimize the energy consumption by lowering both the power and the active time, over all phases of MCU operation. This unique combination of ultra low energy consumption and the performance of the 32-bit ARM Cortex-M3 processor, help designers get more out of the available energy in a variety of applications.

Ultra low energy EFM32G microcontrollers are perfect for:

- Gas metering
- Energy metering
- Water metering
- Smart metering
- Alarm and security systems
- Health and fitness applications
- Industrial and home automation



1.2 EFM32G Development

Because EFM32G use the Cortex-M3 CPU, embedded designers benefit from the largest development ecosystem in the industry, the ARM ecosystem. The development suite spans the whole design process and includes powerful debug tools, and some of the world's top brand compilers. Libraries with documentation and user examples shorten time from idea to market.

The range of EFM32G devices ensure easy migration and feature upgrade possibilities.

Reserved

Registers and bit fields marked with *reserved* are reserved for future use. These should be written to 0 unless otherwise stated in the Register Description. Reserved bits might be read as 1 in future devices.

Reset Value

The reset value denotes the value after reset.

Registers denoted with X have an unknown reset value and need to be initialized before use. Note that, before these registers are initialized, read-modify-write operations might result in undefined register values.

Pin Connections

Pin connections are given as a module prefix followed by a short pin name:

USn_TX (USARTn TX pin)

The pin locations referenced in this document are given in the device-specific datasheet.

2.2 Related Documentation

Further documentation on the EFM32G family and the ARM Cortex-M3 can be found at the Silicon Laboratories and ARM web pages:

www.silabs.com

www.arm.com

- 3× 16-bit Timer/Counter
  - 3 Compare/Capture/PWM channels
  - Dead-Time Insertion on TIMER0
- 16-bit Low Energy Timer
- 24-bit Real-Time Counter
- 3× 8-bit Pulse Counter
  - Asynchronous pulse counting/quadrature decoding
- Watchdog Timer with dedicated RC oscillator @ 50 nA
- **Ultra low power precision analog peripherals**
- 12-bit 1 Msamples/s Analog to Digital Converter
  - 8 input channels and on-chip temperature sensor
  - Single ended or differential operation
  - Conversion tailgating for predictable latency
- 12-bit 500 ksamples/s Digital to Analog Converter
  - 2 single ended channels/1 differential channel
- 2× Analog Comparator
  - Programmable speed/current
  - Capacitive sensing with up to 8 inputs
- Supply Voltage Comparator
- **Ultra efficient Power-on Reset and Brown-Out Detector**
- **2-pin Serial Wire Debug interface**
- 1-pin Serial Wire Viewer
- **Temperature range -40 - 85°C**
- **Single power supply 1.98 - 3.8 V**
- **Packages**
- QFN32
- QFN64
- TQFP48
- TQFP64
- LQFP100
- LFBGA112

3.3 Block Diagram

Figure 3.1 (p. 7) shows the block diagram of EFM32G. The color indicates peripheral availability in the different energy modes, described in Section 3.4 (p. 7) .

Bit	Name	Description
		b1000 Arbitrates after 256 DMA transfers. b1001 Arbitrates after 512 DMA transfers. b1010 - b1111 Arbitrates after 1024 DMA transfers. This means that no arbitration occurs during the DMA transfer because the maximum transfer size is 1024.
[13:4]	n_minus_1	Prior to the DMA cycle commencing, these bits represent the total number of DMA transfers that the DMA cycle contains. You must set these bits according to the size of DMA cycle that you require. The 10-bit value indicates the number of DMA transfers, minus one. The possible values are: b000000000 = 1 DMA transfer b000000001 = 2 DMA transfers b000000010 = 3 DMA transfers b000000011 = 4 DMA transfers b000000100 = 5 DMA transfers . . . b111111111 = 1024 DMA transfers. The controller updates this field immediately prior to it entering the arbitration process. This enables the controller to store the number of outstanding DMA transfers that are necessary to complete the DMA cycle.
[3]	next_useburst	Controls if the chnl_useburst_set [C] bit is set to a 1, when the controller is performing a peripheral scatter-gather and is completing a DMA cycle that uses the alternate data structure. ***Note*** Immediately prior to completion of the DMA cycle that the alternate data structure specifies, the controller sets the chnl_useburst_set [C] bit to 0 if the number of remaining transfers is less than 2^R. The setting of the next_useburst bit controls if the controller performs an additional modification of the chnl_useburst_set [C] bit. In peripheral scatter-gather DMA cycle then after the DMA cycle that uses the alternate data structure completes , either: 0 = the controller does not change the value of the chnl_useburst_set [C] bit. If the chnl_useburst_set [C] bit is 0 then for all the remaining DMA cycles in the peripheral scatter-gather transaction, the controller responds to requests on `dma_req[ ]` and `dma_sreq[ ]`, when it performs a DMA cycle that uses an alternate data structure. 1 = the controller sets the chnl_useburst_set [C] bit to a 1. Therefore, for the remaining DMA cycles in the peripheral scatter-gather transaction, the controller only responds to requests on `dma_req[ ]`, when it performs a DMA cycle that uses an alternate data structure.
[2:0]	cycle_ctrl	The operating mode of the DMA cycle. The modes are: b000 Stop. Indicates that the data structure is invalid. b001 Basic. The controller must receive a new request, prior to it entering the arbitration process, to enable the DMA cycle to complete. b010 Auto-request. The controller automatically inserts a request for the appropriate channel during the arbitration process. This means that the initial request is sufficient to enable the DMA cycle to complete. b011 Ping-pong. The controller performs a DMA cycle using one of the data structures. After the DMA cycle completes, it performs a DMA cycle using the other data structure. After the DMA cycle completes and provided that the host processor has updated the original data structure, it performs a DMA cycle using the original data structure. The controller continues to perform DMA cycles until it either reads an invalid data structure or the host processor changes the cycle_ctrl bits to b001 or b010. See Section 8.4.2.3.4 (p. 47) . b100 Memory scatter/gather. See Section 8.4.2.3.5 (p. 49) . When the controller operates in memory scatter-gather mode, you must only use this value in the primary data structure. b101 Memory scatter/gather. See Section 8.4.2.3.5 (p. 49) . When the controller operates in memory scatter-gather mode, you must only use this value in the alternate data structure. b110 Peripheral scatter/gather. See Section 8.4.2.3.6 (p. 51) .

Bit	Name	Reset	Access	Description
31:4	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
3:0	HFCORECLKDIV	0x0	RW	**HFCORECLK Divider**

Specifies the clock divider for HFCORECLK.

Value	Mode	Description
0	HFCLK	HFCORECLK = HFCLK.
1	HFCLK2	HFCORECLK = HFCLK/2.
2	HFCLK4	HFCORECLK = HFCLK/4.
3	HFCLK8	HFCORECLK = HFCLK/8.
4	HFCLK16	HFCORECLK = HFCLK/16.
5	HFCLK32	HFCORECLK = HFCLK/32.
6	HFCLK64	HFCORECLK = HFCLK/64.
7	HFCLK128	HFCORECLK = HFCLK/128.
8	HFCLK256	HFCORECLK = HFCLK/256.
9	HFCLK512	HFCORECLK = HFCLK/512.

11.5.3 CMU_HFPERCLKDIV - High Frequency Peripheral Clock Division Register

Offset	Bit Position																														
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																								1						0x0		
Access																								RW						RW		
Name																								HFPERCLKEN						HFPERCLKDIV		

Bit	Name	Reset	Access	Description
31:9	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
8	HFPERCLKEN	1	RW	**HFPERCLK Enable**

Set to enable the HFPERCLK.

7:4	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
3:0	HFPERCLKDIV	0x0	RW	**HFPERCLK Divider**

Specifies the clock divider for the HFPERCLK.

Value	Mode	Description
0	HFCLK	HFPERCLK = HFCLK.
1	HFCLK2	HFPERCLK = HFCLK/2.
2	HFCLK4	HFPERCLK = HFCLK/4.
3	HFCLK8	HFPERCLK = HFCLK/8.
4	HFCLK16	HFPERCLK = HFCLK/16.
5	HFCLK32	HFPERCLK = HFCLK/32.
6	HFCLK64	HFPERCLK = HFCLK/64.
7	HFCLK128	HFPERCLK = HFCLK/128.
8	HFCLK256	HFPERCLK = HFCLK/256.
9	HFCLK512	HFPERCLK = HFCLK/512.

11.5.22 CMU_LFBCLKEN0 - Low Frequency B Clock Enable Register 0 (Async Reg)

Offset	Bit Position																															
0x060	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															RW	RW
Name																															LEUART1	LEUART0

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	LEUART1	0	RW	**Low Energy UART 1 Clock Enable**
	Set to enable the clock for LEUART1.			
0	LEUART0	0	RW	**Low Energy UART 0 Clock Enable**
	Set to enable the clock for LEUART0.			

11.5.23 CMU_LFAPRESC0 - Low Frequency A Prescaler Register 0 (Async Reg)

Offset	Bit Position																															
0x068	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																							0x0		0x0			0x0				
Access																							RW		RW			RW				
Name																							LCD		LETIMER0			RTC				

Bit	Name	Reset	Access	Description
31:10	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
9:8	LCD	0x0	RW	**Liquid Crystal Display Controller Prescaler**
	Configure Liquid Crystal Display Controller prescaler			

Value	Mode	Description
0	DIV16	$LFACLK_{LCD}$ = LFACLK/16
1	DIV32	$LFACLK_{LCD}$ = LFACLK/32
2	DIV64	$LFACLK_{LCD}$ = LFACLK/64
3	DIV128	$LFACLK_{LCD}$ = LFACLK/128

7:4	LETIMER0	0x0	RW	**Low Energy Timer 0 Prescaler**
	Configure Low Energy Timer 0 prescaler			

Value	Mode	Description
0	DIV1	$LFACLK_{LETIMER0}$ = LFACLK
1	DIV2	$LFACLK_{LETIMER0}$ = LFACLK/2
2	DIV4	$LFACLK_{LETIMER0}$ = LFACLK/4
3	DIV8	$LFACLK_{LETIMER0}$ = LFACLK/8
4	DIV16	$LFACLK_{LETIMER0}$ = LFACLK/16
5	DIV32	$LFACLK_{LETIMER0}$ = LFACLK/32
6	DIV64	$LFACLK_{LETIMER0}$ = LFACLK/64

13.5.2 PRS_SWLEVEL - Software Level Register

Offset	Bit Position																													
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																									0	0	0	0	0	0	0	0
Access																									RW	RW	RW	RW	RW	RW	RW	RW
Name																									CH7LEVEL	CH6LEVEL	CH5LEVEL	CH4LEVEL	CH3LEVEL	CH2LEVEL	CH1LEVEL	CH0LEVEL

Bit	Name	Reset	Access	Description
31:8	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
7	CH7LEVEL	0	RW	**Channel 7 Software Level**
	See bit 0.			
6	CH6LEVEL	0	RW	**Channel 6 Software Level**
	See bit 0.			
5	CH5LEVEL	0	RW	**Channel 5 Software Level**
	See bit 0.			
4	CH4LEVEL	0	RW	**Channel 4 Software Level**
	See bit 0.			
3	CH3LEVEL	0	RW	**Channel 3 Software Level**
	See bit 0.			
2	CH2LEVEL	0	RW	**Channel 2 Software Level**
	See bit 0.			
1	CH1LEVEL	0	RW	**Channel 1 Software Level**
	See bit 0.			
0	CH0LEVEL	0	RW	**Channel 0 Software Level**
	The value in this register is XOR'ed with the corresponding bit in the SWPULSE register and the selected PRS input signal to generate the channel output.			

13.5.3 PRS_CHx_CTRL - Channel Control Register

Offset	Bit Position																													
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset							0x0				0x00																			0x0		
Access							RW				RW																			RW		
Name							EDSEL				SOURCESEL																			SIGSEL		

Bit	Name	Reset	Access	Description
31:26	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
25:24	EDSEL	0x0	RW	**Edge Detect Select**
	Select edge detection.			

Value	Mode	Description
0	OFF	Signal is left as it is

multiplexed modes. Also for the non-multiplexed 8-bit address mode both the address and data fit into these 16 EBI_AD pins. If more address bits or data bits are needed, external latches can be used to support up to 24-bit addresses or 16-bit data in the multiplexed addressing modes using only the 16 EBI_AD pins.

When a read operation is requested by the Cortex-M3 or DMA via the EBI's AHB interface, the address is transferred onto the EBI_AD bus. After a specific number of cycles, the EBI_REn pin is activated and data is read from the EBI_AD bus. When a write operation is requested, the address is transferred onto the EBI_AD bus and subsequently the write data is transferred onto the EBI_AD bus as the EBI_WEn pin is activated. The detailed operation in the supported modes is presented in the following sections.

In this mode, 8-bit address and 8-bit data is supported. The address is put on the higher 8 bits of the EBI_AD lines while the data uses the lower 8 bits. This mode is set by programming the MODE field in the EBI_CTRL register to D8A8. Read and write signals in 8-bit mode are shown in Figure 14.1 (p. 137) and Figure 14.2 (p. 137) respectively.

Figure 14.1. EBI Non-multiplexed 8-bit Data, 8-bit Address Read Operation

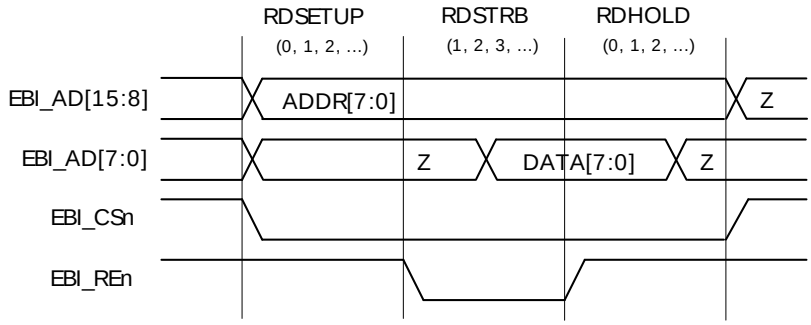
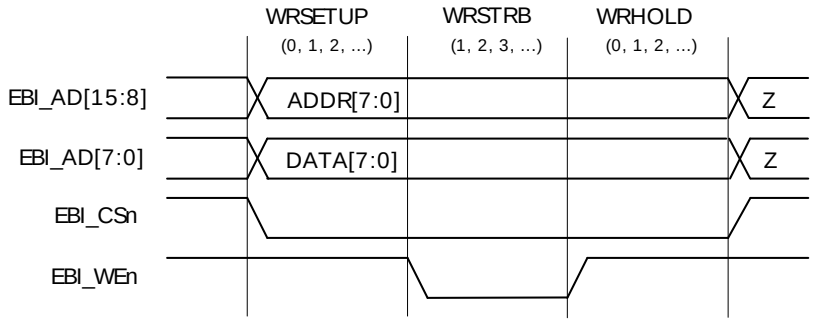


Figure 14.2. EBI Non-multiplexed 8-bit Data, 8-bit Address Write Operation



In this mode, 16-bit address and 16-bit data is supported, but the utilization of an external latch is required. The 16-bit address and 16-bit data bits are multiplexed on the EBI_AD lines. An illustration of such a setup is shown in Figure 14.3 (p. 138) . This mode is set by programming the MODE field in the EBI_CTRL register to D16A16ALE.

Note

In this mode the 16-bit address is organized in 2-byte chunks at memory addresses aligned to 2-byte offsets. Consequently, the LSB of the 16-bit address will always be 0. In order to double the address space, the 16-bit address is internally shifted one bit to the right so that

Bit	Name	Reset	Access	Description
9:8	ADDRHOLD	0x3	RW	**Address Hold Time**
	Sets the number of cycles the address is held after ALE is asserted.			
7:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1:0	ADDRSETUP	0x3	RW	**Address Setup Time**
	Sets the number of cycles the address is driven onto the ADDRDAT bus before ALE is asserted. If set to 0, 1 cycle is inserted by HW.			

14.5.8 EBI_RDTIMING1 - Read Timing Register 1

Offset	Bit Position																														
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset		0	0	0											0x3				0x3F												0x3	
Access		RW	RW	RW											RW				RW												RW	
Name		PAGEMODE	PREFETCH	HALFRE											RDHOLD				RDSTRB												RDSETUP	

Bit	Name	Reset	Access	Description
31	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
30	PAGEMODE	0	RW	**Page Mode Access Enable**
	Enables or disables page mode reads.			
29	PREFETCH	0	RW	**Prefetch Enable**
	Enables or disables prefetching of data from sequential address.			
28	HALFRE	0	RW	**Half Cycle REn Strobe Duration Enable**
	Enables or disables half cycle duration of the REn strobe in the last RDSTRB cycle.			
27:18	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
17:16	RDHOLD	0x3	RW	**Read Hold Time**
	Sets the number of cycles CSn is held active after the REn is deasserted. This interval is used for bus turnaround.			
15:14	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
13:8	RDSTRB	0x3F	RW	**Read Strobe Time**
	Sets the number of cycles the REn is held active. After the specified number of cycles, data is read. If set to 0, 1 cycle is inserted by HW.			
7:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1:0	RDSETUP	0x3	RW	**Read Setup Time**
	Sets the number of cycles the address setup before REn is asserted.			

14.5.9 EBI_WRTIMING1 - Write Timing Register 1

Offset	Bit Position																														
0x020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset			0	0											0x3				0x3F												0x3	
Access			RW	RW											RW				RW												RW	
Name			WBUFDIS	HALFWE											WRHOLD				WRSTRB												WRSETUP	

Bit	Name	Reset	Access	Description
31:30	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
29	WBUFDIS	0	RW	**Write Buffer Disable**
	Enables or disables the write buffer.			
28	HALFWE	0	RW	**Half Cycle WEn Strobe Duration Enable**
	Enables or disables half cycle duration of the WEn strobe in the last WRSTRB cycle.			
27:18	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
17:16	WRHOLD	0x3	RW	**Write Hold Time**
	Sets the number of cycles CSn is held active after the WEn is deasserted.			
15:14	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
13:8	WRSTRB	0x3F	RW	**Write Strobe Time**
	Sets the number of cycles the WEn is held active. If set to 0, 1 cycle is inserted by HW.			
7:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1:0	WRSETUP	0x3	RW	**Write Setup Time**
	Sets the number of cycles the address setup before WEn is asserted.			

14.5.10 EBI_POLARITY1 - Polarity Register 1

Offset	Bit Position																													
0x024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																											0	0	0	0	0	0
Access																											RW	RW	RW	RW	RW	RW
Name																											BLPOL	ARDYPOL	ALEPOL	WEPOL	REPOL	CSPOL

Bit	Name	Reset	Access	Description
31:6	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
5	BLPOL	0	RW	**BL Polarity**
	Sets the polarity of the EBI_BLn lines.			

Value	Mode	Description
0	ACTIVELOW	BLn[1:0] are active low.
1	ACTIVEHIGH	BLn[1:0] are active high.

4	ARDYPOL	0	RW	**ARDY Polarity**
	Sets the polarity of the EBI_ARDY line.			

Value	Mode	Description
0	ACTIVELOW	ARDY is active low.
1	ACTIVEHIGH	ARDY is active high.

3	ALEPOL	0	RW	**Address Latch Polarity**
	Sets the polarity of the EBI_ALE line.			

Value	Mode	Description
0	ACTIVELOW	ALE is active low.
1	ACTIVEHIGH	ALE is active high.

2	WEPOL	0	RW	**Write Enable Polarity**
	Sets the polarity of the EBI_WEn and EBI_NANDWEn lines.			

Bit	Name	Reset	Access	Description
31:3	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
2	ECCCLEAR	0	W1	**Error Correction Code Clear**
	Write to 1 to clear ECCPARITY.			
1	ECCSTOP	0	W1	**Error Correction Code Generation Stop**
	Write to 1 to stop ECC generation.			
0	ECCSTART	0	W1	**Error Correction Code Generation Start**
	Write to 1 to start ECC generation.			

14.5.22 EBI_STATUS - Status Register

Offset	Bit Position																														
0x054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																			0	0		0	0	0				0				0
Access																			R	R		R	R	R				R				R
Name																			TFTDDEMPTY	DDACT		TFTPIXELFULL	TFTPIXEL1EMPTY	TFTPIXEL0EMPTY				ECCACT				AHBACT

Bit	Name	Reset	Access	Description
31:14	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
13	TFTDDEMPTY	0	R	**EBI_TFTDD register is empty.**
	Indicates that EBI_TFTDD register is empty.			
12	DDACT	0	R	**EBI Busy with Direct Drive Transactions.**
	Indicates that EBI is busy with Direct Drive Transactions.			
11	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
10	TFTPIXELFULL	0	R	**EBI_TFTPIXEL0 is full.**
	Indicates that EBI_TFTPIXEL is full.			
9	TFTPIXEL1EMPTY	0	R	**EBI_TFTPIXEL1 is empty.**
	Indicates that EBI_TFTPIXEL1 is empty.			
8	TFTPIXEL0EMPTY	0	R	**EBI_TFTPIXEL0 is empty.**
	Indicates that EBI_TFTPIXEL0 is empty.			
7:5	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
4	ECCACT	0	R	**EBI ECC Generation Active.**
	Indicates that EBI is generating ECC.			
3:1	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
0	AHBACT	0	R	**EBI Busy with AHB Transaction.**
	Indicates that EBI is busy with an AHB Transaction.			

I2Cn_STAT	Description	I2Cn_IF	Required interaction	Response
			START	START will be sent when bus becomes idle
-	Arbitration lost	ARBLOST interrupt flag	None	
			START	START will be sent when bus becomes idle

15.3.8 Bus States

The I2Cn_STATE register can be used to determine which state the I^{2}C module and the I^{2}C bus are in at a given time. The register consists of the STATE bit-field, which shows which state the I^{2}C module is at in any ongoing transmission, and a set of single-bits, which reveal the transmission mode, whether the bus is busy or idle, and whether the bus is held by this I^{2}C module waiting for a software response.

The possible values of the STATE field are summarized in Table 15.7 (p. 187) . When this field is cleared, the I^{2}C module is not a part of any ongoing transmission. The remaining status bits in the I2Cn_STATE register are listed in Table 15.8 (p. 187) .

Table 15.7. I^{2}C STATE Values

Mode	Value	Description
IDLE	0	No transmission is being performed by this module.
WAIT	1	Waiting for idle. Will send a start condition as soon as the bus is idle.
START	2	Start being transmitted
ADDR	3	Address being transmitted or has been received
ADDRACK	4	Address ACK/NACK being transmitted or received
DATA	5	Data being transmitted or received
DATAACK	6	Data ACK/NACK being transmitted or received

Table 15.8. I^{2}C Transmission Status

Bit	Description
BUSY	Set whenever there is activity on the bus. Whether or not this module is responsible for the activity cannot be determined by this byte.
MASTER	Set when operating as a master. Cleared at all other times.
TRANSMITTER	Set when operating as a transmitter; either a master transmitter or a slave transmitter. Cleared at all other times
BUSHOLD	Set when the bus is held by this I^{2}C module because an action is required by software.
NACK	Only valid when bus is held and STATE is ADDRACK or DATAACK. In that case it is set if a NACK was received. In all other cases, the bit is cleared.

Note

I2Cn_STATE reflects the internal state of the I^{2}C module, and therefore only held constant as long as the bus is held, i.e. as long as BUSHOLD in I2Cn_STATUS is set.

15.3.9 Slave Operation

The I^{2}C module operates in master mode by default. To enable slave operation, i.e. to allow the device to be addressed as an I^{2}C slave, the SLAVE bit in I2Cn_CTRL must be set. In this case the slave operates in a mixed mode, both capable of starting transmissions as a master, and being addressed as a slave. When operating in the slave mode, HFPERCLK frequency must be higher than 4.2 MHz for Standard-mode, 11 MHz for Fast-mode, and 24.4 MHz for Fast-mode Plus.

15.4 Register Map

The offset register address is relative to the registers base address.

Offset	Name	Type	Description
0x000	I2Cn_CTRL	RW	Control Register
0x004	I2Cn_CMD	W1	Command Register
0x008	I2Cn_STATE	R	State Register
0x00C	I2Cn_STATUS	R	Status Register
0x010	I2Cn_CLKDIV	RW	Clock Division Register
0x014	I2Cn_SADDR	RW	Slave Address Register
0x018	I2Cn_SADDRMASK	RW	Slave Address Mask Register
0x01C	I2Cn_RXDATA	R	Receive Buffer Data Register
0x020	I2Cn_RXDATAP	R	Receive Buffer Data Peek Register
0x024	I2Cn_TXDATA	W	Transmit Buffer Data Register
0x028	I2Cn_IF	R	Interrupt Flag Register
0x02C	I2Cn_IFS	W1	Interrupt Flag Set Register
0x030	I2Cn_IFC	W1	Interrupt Flag Clear Register
0x034	I2Cn_IEN	RW	Interrupt Enable Register
0x038	I2Cn_ROUTE	RW	I/O Routing Register

15.5 Register Description

15.5.1 I2Cn_CTRL - Control Register

Offset	Bit Position																													
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset														0x0			0		0x0				0x0			0	0	0	0	0	0	0
Access														RW			RW		RW				RW			RW	RW	RW	RW	RW	RW	RW
Name														CLTO			GIBITO		BITO				CLHR			GCAMEN	ARBDIS	AUTOSN	AUTOSE	AUTOACK	SLAVE	EN

Bit	Name	Reset	Access	Description
31:19	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
18:16	CLTO	0x0	RW	**Clock Low Timeout**

Use to generate a timeout when CLK has been low for the given amount of time. Wraps around and continues counting when the timeout is reached.

Value	Mode	Description
0	OFF	Timeout disabled
1	40PCC	Timeout after 40 prescaled clock cycles. In standard mode at 100 kHz, this results in a 50us timeout.
2	80PCC	Timeout after 80 prescaled clock cycles. In standard mode at 100 kHz, this results in a 100us timeout.
3	160PCC	Timeout after 160 prescaled clock cycles. In standard mode at 100 kHz, this results in a 200us timeout.
4	320PPC	Timeout after 320 prescaled clock cycles. In standard mode at 100 kHz, this results in a 400us timeout.
5	1024PPC	Timeout after 1024 prescaled clock cycles. In standard mode at 100 kHz, this results in a 1280us timeout.

15	GIBITO	0	RW	**Go Idle on Bus Idle Timeout**

Table 16.7. USART Baud Rates @ 4MHz Peripheral Clock

Desired baud rate [baud/s]	USARTn_OVS =00			USARTn_OVS =01		
	USARTn_CLKDIV/256	Actual baud rate [baud/s]	Error %	USARTn_CLKDIV/256	Actual baud rate [baud/s]	Error %
600	415,75	599,88	-0,02	832,25	600,06	0,01
1200	207,25	1200,48	0,04	415,75	1199,76	-0,02
2400	103,25	2398,082	-0,08	207,25	2400,96	0,04
4800	51	4807,692	0,16	103,25	4796,163	-0,08
9600	25	9615,385	0,16	51	9615,385	0,16
14400	16,25	14492,75	0,64	33,75	14388,49	-0,08
19200	12	19230,77	0,16	25	19230,77	0,16
28800	7,75	28571,43	-0,79	16,25	28985,51	0,64
38400	5,5	38461,54	0,16	12	38461,54	0,16
57600	3,25	58823,53	2,12	7,75	57142,86	-0,79
76800	2,25	76923,08	0,16	5,5	76923,08	0,16
115200	1,25	111111,1	-3,55	3,25	117647,1	2,12
230400	0	250000	8,51	1,25	222222,2	-3,55

16.3.2.3 Data Transmission

Asynchronous data transmission is initiated by writing data to the transmit buffer using one of the methods described in Section 16.3.2.3.1 (p. 212) . When the transmission shift register is empty and ready for new data, a frame from the transmit buffer is loaded into the shift register, and if the transmitter is enabled, transmission begins. When the frame has been transmitted, a new frame is loaded into the shift register if available, and transmission continues. If the transmit buffer is empty, the transmitter goes to an idle state, waiting for a new frame to become available.

Transmission is enabled through the command register USARTn_CMD by setting TXEN, and disabled by setting TXDIS in the same command register. When the transmitter is disabled using TXDIS, any ongoing transmission is aborted, and any frame currently being transmitted is discarded. When disabled, the TX output goes to an idle state, which by default is a high value. Whether or not the transmitter is enabled at a given time can be read from TXENS in USARTn_STATUS.

When the USART transmitter is enabled and there is no data in the transmit shift register or transmit buffer, the TXC flag in USARTn_STATUS and the TXC interrupt flag in USARTn_IF are set, signaling that the transmitter is idle. The TXC status flag is cleared when a new frame becomes available for transmission, but the TXC interrupt flag must be cleared by software.

16.3.2.3.1 Transmit Buffer Operation

The transmit-buffer is a 2-level FIFO buffer. A frame can be loaded into the buffer by writing to USARTn_TXDATA, USARTn_TXDATAX, USARTn_TXDOUBLE or USARTn_TXDOUBLEX. Using USARTn_TXDATA allows 8 bits to be written to the buffer, while using USARTn_TXDOUBLE will write 2 frames of 8 bits to the buffer. If 9-bit frames are used, the 9th bit of the frames will in these cases be set to the value of BIT8DV in USARTn_CTRL.

To set the 9th bit directly and/or use transmission control, USARTn_TXDATAX and USARTn_TXDOUBLEX must be used. USARTn_TXDATAX allows 9 data bits to be written, as well as a set of control bits regarding the transmission of the written frame. Every frame in the buffer is stored with 9 data bits and additional transmission control bits. USARTn_TXDOUBLEX allows two

Bit	Name	Reset	Access	Description
31:16	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
15	FERR	0	R	**Receive Data Framing Error**
	Set if data in buffer has a framing error. Can be the result of a break condition.			
14	PERR	0	R	**Receive Data Parity Error**
	Set if data in buffer has a parity error.			
13:9	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
8:0	RXDATA	0x000	R	**RX Data**
	Use this register to access data read from the LEUART. Buffer is cleared on read access.			

18.5.8 LEUARTn_RXDATA - Receive Buffer Data Register

Offset	Bit Position																													
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																									0x00							
Access																									R							
Name																									RXDATA							

Bit	Name	Reset	Access	Description
31:8	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
7:0	RXDATA	0x00	R	**RX Data**
	Use this register to access data read from LEUART. Buffer is cleared on read access. Only the 8 LSB can be read using this register.			

18.5.9 LEUARTn_RXDATAXP - Receive Buffer Data Extended Peek Register

Offset	Bit Position																													
0x020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																	0	0						0x000								
Access																	R	R						R								
Name																	FERRP	PERRP						RXDATAP								

Bit	Name	Reset	Access	Description
31:16	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
15	FERRP	0	R	**Receive Data Framing Error Peek**
	Set if data in buffer has a framing error. Can be the result of a break condition.			
14	PERRP	0	R	**Receive Data Parity Error Peek**
	Set if data in buffer has a parity error.			

Bit	Name	Reset	Access	Description

Value	Mode	Description
0	RISING	Rising edges detected
1	FALLING	Falling edges detected
2	BOTH	Both edges detected
3	NONE	No edge detection, signal is left as it is

23:22 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

21 FILT 0 RW **Digital Filter**

Enable digital filter.

Value	Mode	Description
0	DISABLE	Digital filter disabled
1	ENABLE	Digital filter enabled

20 INSEL 0 RW **Input Selection**

Select Compare/Capture channel input.

Value	Mode	Description
0	PIN	TIMERnCCx pin is selected
1	PRS	PRS input (selected by PRSSEL) is selected

19 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

18:16 PRSSEL 0x0 RW **Compare/Capture Channel PRS Input Channel Selection**

Select PRS input channel for Compare/Capture channel.

Value	Mode	Description
0	PRSCH0	PRS Channel 0 selected as input
1	PRSCH1	PRS Channel 1 selected as input
2	PRSCH2	PRS Channel 2 selected as input
3	PRSCH3	PRS Channel 3 selected as input
4	PRSCH4	PRS Channel 4 selected as input
5	PRSCH5	PRS Channel 5 selected as input
6	PRSCH6	PRS Channel 6 selected as input
7	PRSCH7	PRS Channel 7 selected as input

15:14 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

13:12 CUFOA 0x0 RW **Counter Underflow Output Action**

Select output action on counter underflow.

Value	Mode	Description
0	NONE	No action on counter underflow
1	TOGGLE	Toggle output on counter underflow
2	CLEAR	Clear output on counter underflow
3	SET	Set output on counter underflow

11:10 COFOA 0x0 RW **Counter Overflow Output Action**

Select output action on counter overflow.

Value	Mode	Description
0	NONE	No action on counter overflow
1	TOGGLE	Toggle output on counter overflow
2	CLEAR	Clear output on counter overflow
3	SET	Set output on counter overflow

9:8 CMOA 0x0 RW **Compare Match Output Action**

Select output action on compare match.

Value	Mode	Description
0	NONE	No action on compare match
1	TOGGLE	Toggle output on compare match
2	CLEAR	Clear output on compare match

21.3.5.2 Continuous Output Generation

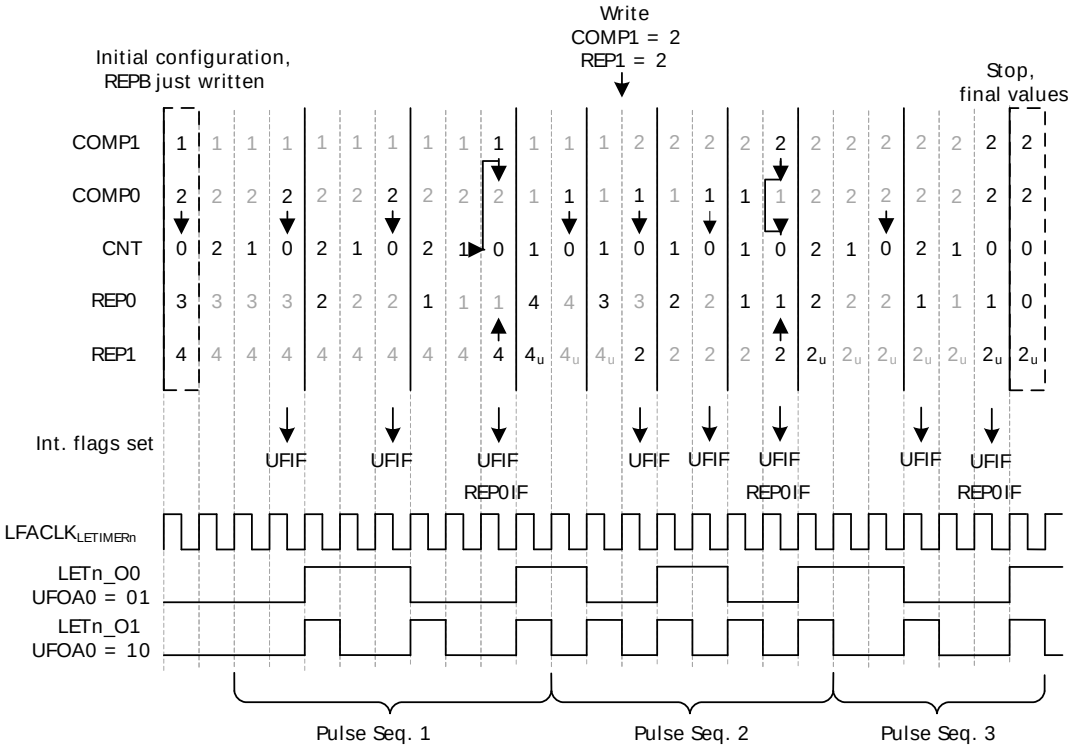
Example 21.2. LETIMER Continuous Output Generation

In some scenarios, it might be desired to make LETIMER generate a continuous waveform. Very simple constant waveforms can be generated without the repeat counter as shown in Figure 21.6 (p. 327), but to generate changing waveforms, using the repeat counter and buffer registers can prove advantageous.

For the example in Figure 21.10 (p. 329), the goal is to produce a pulse train consisting of 3 sequences with the following properties:

- 3 pulses with periods of 3 cycles
- 4 pulses with periods of 2 cycles
- 2 pulses with periods of 3 cycles

Figure 21.10. LETIMER Continuous Operation



The first two sequences are loaded into the LETIMER before the timer is started.

LETIMERn_COMP0 is set to 2 (cycles – 1), and LETIMERn_REP0 is set to 3 for the first sequence, and the second sequence is loaded into the buffer registers, i.e. COMP1 is set to 1 and LETIMERn_REP1 is set to 4.

The LETIMER is set to trigger an interrupt when LETIMERn_REP0 is done by setting REP0 in LETIMERn_IEN. This interrupt is a good place to update the values of the buffers. Last but not least REPMODE in LETIMERn_CTRL is set to buffered mode, and the timer is started.

In the interrupt routine the buffers are updated with the values for the third sequence. If this had not been done, the timer would have stopped after the second sequence.

The final result is shown in Figure 21.10 (p. 329). The pulse output is grouped to show which sequence generated which output. Toggle output is also shown in the figure. Note that the toggle output is not aligned with the pulse outputs.

Note

23.3.2 Response Time

There is a delay from when the actual input voltage changes polarity, to when the output toggles. This period is called the response time and can be altered by increasing or decreasing the bias current to the comparator through the BIASPROG, FULLBIASPROG and HALFBIAS fields in the ACMPn_CTRL register, as illustrated in Table 23.1 (p. 356) Setting the HALFBIAS bit in ACMPn_CTRL effectively halves the current. Setting a lower bias current will result in lower power consumption, but a longer response time.

If the FULLBIAS bit is set, the highest hysteresis level should be used to avoid glitches on the output.

Table 23.1. Bias Configuration

BIASPROG	Bias Current (µA), HYSTSEL=0			
	FULLBIAS=0, HALFBIAS=1	FULLBIAS=0, HALFBIAS=0	FULLBIAS=1, HALFBIAS=1	FULLBIAS=1, HALFBIAS=0
0b0000	0.05	0.1	3.3	6.5
0b0001	0.1	0.2	6.5	13
0b0010	0.2	0.4	13	26
0b0011	0.3	0.6	20	39
0b0100	0.4	0.8	26	52
0b0101	0.5	1.0	33	65
0b0110	0.6	1.2	39	78
0b0111	0.7	1.4	46	91
0b1000	1.0	2.0	65	130
0b1001	1.1	2.2	72	143
0b1010	1.2	2.4	78	156
0b1011	1.3	2.6	85	169
0b1100	1.4	2.8	91	182
0b1101	1.5	3.0	98	195
0b1110	1.6	3.2	104	208
0b1111	1.7	3.4	111	221

23.3.3 Hysteresis

In the analog comparator, hysteresis can be configured to 8 different levels, including off which is level 0, through the HYSTSEL field in ACMPn_CTRL. When the hysteresis level is set above 0, the digital output will not toggle until the positive input voltage is at a voltage equal to the hysteresis level above or below the negative input voltage (see Figure 23.2 (p. 357)). This feature can be used to filter out uninteresting input fluctuations around zero and only show changes that are big enough to breach the hysteresis threshold. Note that the ACMP current consumption will be influenced by the selected hysteresis level and in general decrease with increasing HYSTSEL values.

Bit	Name	Reset	Access	Description

Value	Mode	Description
7	128CYCLES	128 ADC_CLK cycles acquisition time for single sample
8	256CYCLES	256 ADC_CLK cycles acquisition time for single sample

19 Reserved — *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

18:16 REF 0x0 RW **Single Sample Reference Selection**

Select reference to ADC single sample mode.

Value	Mode	Description
0	1V25	Internal 1.25 V reference
1	2V5	Internal 2.5 V reference
2	VDD	Buffered VDD
3	5VDIFF	Internal differential 5 V reference
4	EXTSINGLE	Single ended external reference from pin 6
5	2XEXTDIFF	Differential external reference, 2x(pin 6 - pin 7)
6	2XVDD	Unbuffered 2xVDD

15:12 Reserved — *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

11:8 INPUTSEL 0x0 RW **Single Sample Input Selection**

Select input to ADC single sample mode in either single ended mode or differential mode.

DIFF = 0		
Mode	Value	Description
CH0	0	ADCn_CH0
CH1	1	ADCn_CH1
CH2	2	ADCn_CH2
CH3	3	ADCn_CH3
CH4	4	ADCn_CH4
CH5	5	ADCn_CH5
CH6	6	ADCn_CH6
CH7	7	ADCn_CH7
TEMP	8	Temperature reference
VDDDIV3	9	VDD/3
VDD	10	VDD
VSS	11	VSS
VREFDIV2	12	VREF/2
DAC0OUT0	13	DAC0 output 0
DAC0OUT1	14	DAC0 output 1
DIFF = 1		
Mode	Value	Description
CH0CH1	0	Positive input: ADCn_CH0 Negative input: ADCn_CH1
CH2CH3	1	Positive input: ADCn_CH2 Negative input: ADCn_CH3
CH4CH5	2	Positive input: ADCn_CH4 Negative input: ADCn_CH5
CH6CH7	3	Positive input: ADCn_CH6 Negative input: ADCn_CH7
DIFF0	4	Differential 0 (Short between positive and negative inputs)

7:6 Reserved — *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

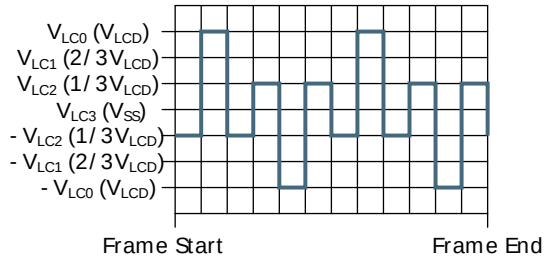
5:4 RES 0x0 RW **Single Sample Resolution Select**

Select single sample conversion resolution.

Value	Mode	Description
0	12BIT	12-bit resolution
1	8BIT	8-bit resolution
2	6BIT	6-bit resolution
3	OVS	Oversampling enabled. Oversampling rate is set in OVSRSEL

3 Reserved — *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

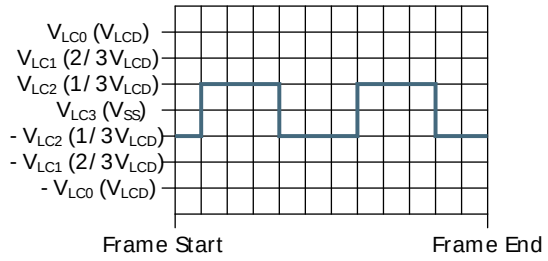
Figure 29.31. LCD 1/3 Bias and Triplex Multiplexing - LCD_SEG0-LCD_COM1



1/3 bias and triplex multiplexing - LCD_SEG0-LCD_COM2

- DC voltage = 0 (over one frame)
- $V_{RMS} = 0.33 \times V_{LCD\_OUT}$
- The LCD display pixel that is connected to LCD_SEG0 and LCD_COM2 will be OFF with this waveform

Figure 29.32. LCD 1/3 Bias and Triplex Multiplexing - LCD_SEG0-LCD_COM2



29.3.3.6 Waveforms with 1/3 Bias and Quadruplex Multiplexing

In this mode, each frame is divided into 8 periods. All COM lines can be multiplexed with all segment lines. Figures show 1/3 bias and quadruplex multiplexing (waveforms show two frames).

Figure 29.33. LCD 1/3 Bias and Quadruplex Multiplexing - LCD_COM0